
Release Notes

Be Operating System, Developer Releases 8 and 8.1

These notes describe changes that have been made to the Be Operating System (BeOS) for Developer Release 8, and the additions and bug fixes that you'll find in Release 8.1. The 8.1 amendments are listed first, followed by the changes from Release 7 to Release 8.

For more information about Developer Releases 8 and 8.1, including a new edition of *The Be Book*, check the developer area of the Be Web site, <http://www.be.com>.

Release 8.1 Additions and Bug Fixes

Among that which is new in Release 8.1, the following fixes and features are notable.

New Graphics Card Support

The BeOS now supports the Diamond Stealth 64 Video 2001 card. This card is based on the S3 Trio 64 V+ chip set.

Note: Regard the “Video” in the Diamond card’s name—don’t confuse this card with the Stealth 64 “Graphics” card, which we *don’t* yet support. Also, the Stealth 64 Video card itself is marked as “2001”; the box it comes in may be marked “2201”.

Changes have been made to begin support for graphics cards with 4 megabytes of VRAM. *However, these changes haven’t been fully tested.* If you’ve got a 4 megabyte card lying around, feel free to try it; but don’t rush out and buy a new card on the basis of this incipient support.

BeMail and the Be Mail Daemon

A number of bugs and inconveniences in the BeMail application and Be’s **mail_daemon** program have been corrected. Among other fixes:

- The mail system no longer balks at content-less messages.

- BeMail no longer clobbers a message's "Status" field after the field has been edited.
- "Sent" mail messages are no longer spuriously deleted.
- SMTP support has been expanded to work with Apple's Internet Mail Server.

Some features have been added to BeMail and the E-mail preferences app, among which:

- The E-mail preferences now lets you set the "Real Name" and "Reply To" values for your POP account.
- BeMail lets you set its font, and define and apply your own mail signature (see the application's Preferences menu item). Furthermore, the message editor understands the following emacs-style control-key bindings:

<u>Key</u>	<u>Effect</u>
a	Move the cursor to the start of the current line.
e	Move the cursor to the end of the line.
f	Move the cursor forward one character.
b	Move the cursor back one character.
n	Move to the next line.
p	Move to the previous line.
o	"Open" the current line—insert a new blank line.
d	Delete the current character.
k	"Kill" the line—delete to the end of the current line.
y	Insert the previously killed text.

Media Kit Buffer Allocation

Due to a bug in the Media Kit's buffer allocation scheme, audio subscribers were (sometimes) prevented from using non-default sized sound buffers. This bug has been fixed; you can now set and re-set the size of the audio buffers as often as you wish, and to any size you want.

SIM Add-ons for CAM Devices

The kernel now recognizes the SCSI CAM SIMs that you add (as add-ons) to the `/system/add-ons/kernel/cam` directory.

Contiguous Physical Memory and Exported Functions for Areas

A new area-locking constant, `B_CONTIGUOUS`, has been added to the Kernel Kit. You use the constant as an argument to `create_area()`. When you create a `B_CONTIGUOUS` area, the pages of memory for the area are immediately allocated and locked into RAM, and are

guaranteed to be (physically) contiguous. Contiguous physical memory is needed by certain devices; the new locking constant is meant to be used by drivers for these devices.

Also, the Kernel Kit now exports all the area functions (defined in `<kernel/OS.h>`) so they can be used by device drivers.

24-Bit Bitmap Printing

You can now print bitmaps that are defined in 24-bit color space (as a `color_space` constant, the space is known as `B_RGB_32_BIT`).

PCI Memory Space Limit

The PCI memory space has been increased to 128 megabytes. The pre-8.1 limit was 16 megabytes.

Standard C Library String Functions

Imperfections in some of the standard C string-parsing functions (such as `fscanf()`) have been corrected.

Scrolling in a BTextView

BTextView's `ScrollToSelection()` function has been fixed. You should now be able to scroll to the selection regardless of where it is.

PPP is Feeling Much Better These Days

PPP is able to connect successfully to many more Internet Service Providers, most notably **ibm.net**.

Front Panel LEDs

The pulsing LEDs in some editions of the BeBox didn't work properly (or at all). This has been fixed.

Principal Changes from Release 7 to Release 8

Developer Release 8 of the Be Operating System provides two new kits and a handful of architectural changes.

The 3D Kit

The eagerly awaited 3D Kit is here. You get headers files, source code examples, a demo application, and, on the Web site, a “Welcome to the 3D World” white paper. What you don’t get, for now, is entertaining technical documentation. But keep listening; we’ll publish documentation, more sample code, and some Kit-clarifying apps on the Web site as soon as we can.

For pointers to the source code examples and the existing Web site white paper, see “The 3D Kit” chapter of *The Be Book*.

The Game Kit

This release introduces the Game Kit, a collection of software designed especially for game applications. Currently, the Kit consists of just one class: BWindowScreen. A BWindowScreen object bypasses the Application Server to provide direct access to the graphics card driver for the screen. It lets an application take over the entire screen within a workspace, set up a game-specific graphics environment, call driver-implemented drawing functions, configure the frame buffer and color palette, and scroll the display area that’s shown on-screen within the larger area defined by the frame buffer.

A BWindowScreen object is now the only way you can get direct access to the screen. BWindowScreen functions usurp the Interface Kit’s **lock_screen()**, **unlock_screen()**, and **get_screen_info()** functions.

Libraries

The **libpos.so** library has been incorporated into **libbe.so**. Because of this, all applications and programs—except for those that are purely Posix—must construct a BApplication object.

Printing

Printing arrives with this release in the form of the Print Server and the new BPrintJob class in the Interface Kit. See “The Interface Kit” chapter in *The Be Book* for details. An API for printer drivers will be added in the next release.

Run-Time Information

The run-time class information system, which relied on the BClassInfo class, has been replaced by a system that’s integrated into the C++ language. The old macros still work as does the **inherited** keyword.

Messaging

A handful of changes have been made to the messaging mechanism defined in the Application and Interface Kits:

- The new **BMessageFilter** class lets **BLooper** and **BHandler** objects easily filter the messages that they receive. Messages can be filtered based on their command constants, how they're delivered, and whether they have a remote or local source. See the **BMessageFilter** class in “The Application Kit” chapter of *The Be Book* for details.

This new class obsoletes the four filtering functions defined in the **BWindow** class: **FilterMouseDown()**, **FilterKeyDown()**, **FilterMouseMoved()**, and **FilterMessageDropped()**. Although the **BWindow** functions will still work in DR8, they'll be removed in the next release.

- In previous releases, messages that the user dragged and dropped on a window were announced by a **B_MESSAGE_DROPPED** system message that wrapped around the dropped message. The wrapper message is now history. Beginning with this release, dropped messages are dispatched just like messages that are delivered programmatically (by being posted or sent). Instead of implementing a special **MessageDropped()** function to receive and respond to dropped messages, you can include them in your implementation of **MessageReceived()**.

The old system distinguished dropped messages from messages received in other ways because being the destination of a drag-and-drop operation can involve obvious graphical obligations. This hasn't changed. However, now you can ask a **BMessage** object how it was delivered. The **BMessage** class has two new functions—**WasDropped()** and **DropPoint()**—that provide the same information you got from the old **MessageDropped()** function, but now you get it from the dropped message itself.

- Previously, the dispatching machinery deferred to system messages. If you posted a message that matched a system message and named a target **BHandler** for it, the dispatcher would ignore the target you named and dispatch it like any other system message. For example, if you posted a **B_KEY_DOWN** message and named a **BView** target that wasn't the focus view, the focus view would get the message anyway. Now, however, dispatching defers to the target you name (or the target view where the user drops the message). The dispatcher always calls a function of the target **BHandler**. The introduction to “The Application Kit” chapter in *The Be Book* discusses the new system in more detail.
- **BHandler** objects can now be chained together in a linked list. The **BHandler** class defines new **SetNextHandler()** and **NextHandler()** functions and its implementation of **MessageReceived()** passes an unrecognized message to the next handler. To get this behavior, derived classes should be sure to call the inherited version of **MessageReceived()**. (By default, the next handler for a **BView** object is its parent view—or, if it's at the top of the view hierarchy, the **BWindow** object.)

- A **DispatchMessage()** function you define no longer has to lock the BLooper; the object is locked before **DispatchMessage()** is called.

Keyboard Navigation

This release introduces a new system of keyboard navigation. Users can press the Tab key to move the focus of keyboard actions from view to view, and can operate the view currently in focus by pressing other keys (typically the space bar and arrow keys).

The Interface Kit takes care of the mechanics of between-view navigation. The classes you derive from BView must implement the hook functions that respond to keyboard actions (**KeyDown()**) and changes to the focus view (**MakeFocus()**). See “Keyboard Navigation” in *The Be Book* overview to the BView class for details.

Keyboard navigation introduces a new BView flag, **B_NAVIGABLE**. When set, it’s possible for the user to navigate to the view; when not set (for example, when the view is disabled), navigation will bypass the view.

The Interface Kit makes the BListView class, the new BMenuField class, and all classes derived from BControl navigable. The BTextControl class relinquishes its private navigation system so that it can participate in the more general mechanism.

Graphics Parameters

You can now set a BView’s graphics parameters at any time—for example, when constructing the object. Previously, you had to wait until the object was attached to a window. It’s still the case that the parameters become alive only when the BView is placed in a window. That’s when the Application Server becomes aware of the view and establishes its graphics environment. Now, however, a BView caches the value that’s set before it belongs to a window and delivers it to the Server when it becomes attached.

API Changes to the Software Kits

This section lists the API changes, small and large, that have been made to previously released software kits.

The Application Kit

Most of the changes to the Application Kit affect the system of sending and receiving messages, which was discussed above. In addition:

- The arguments passed to BClipboard’s **FindData()** function have been juggled to match the order of arguments passed to the identically named BMessage function.

It's now *type, index, numBytes*, instead of *type, numBytes, index*. The *index* is still optional.

- In addition to **WasDropped()** and **DropPoint()** mentioned above, the BMessage class has added a **IsSourceRemote()** function. For compatibility, **IsSenderWaiting()** is now called **IsSourceWaiting()**.
- The BMessenger class has new **IsValid()** and **Team()** functions.
- The BMessageQueue class no longer has the version of **RemoveMessage()** that would remove a class of messages. You must now remove them one at a time
- BLooper's **CheckLock()** function is now called **IsLocked()**.

The Storage Kit

Not much new in the Storage Kit—as we all should know by now, the file system part of the Kit (at least) is headed for fairly extensive surgery come DR9. So it's resting until then.

The one API addition to the Kit is in the BFile class. The class now boasts a **copy_status_hook()** function protocol. This protocol is intended to typify a “copy status” function that you can (optionally) pass as an argument to the **CopyTo()** function. Thus primed, the **CopyTo()** function calls back to your “copy status” function as it is copying your file. Through these call backs you can implement (for example) a GUI copy status “fuel gauge.” See the BFile class for details.

Interface Kit

The Interface Kit has three new classes: BPrintJob, BColorControl, and BMenuField. A BPrintJob object manages an application's interaction with the Print Server. The application creates the object when the user asks to print something and deletes it when the print job is finished. A BColorControl object lets the user pick a color, and a BMenuField controls a labeled pop-up menu. All three classes are documented in the “The Interface Kit” chapter of *The Be Book*.

Here are the other changes to the Kit API:

- The BView class has an **IsPrinting()** function that reports whether or not the drawing a BView is engaged in is destined for the printer.
- The BView class has added an **AllAttached()** hook function to go with **AttachedToWindow()**. The new function is called after all BViews being added to a window have received the **AttachedToWindow()** notification. There are also two similar hook functions—**DetachedFromWindow()** and **AllDetached()**—that are called when a BView is removed from a window.

- Two classes, BBox and BStringView, have lost their **AttachedToWindow()** functions. Their constructors now set up default values for graphic parameters.
- Several kinds of objects now set up their background colors to match the background colors of their parent views. Two classes, BCheckBox and BRadioButton, have added **AttachedToWindow()** functions for this purpose.
- A BView now knows whether it's the target of a scroll bar. Its **ScrollBar()** function returns the BScrollBar objects that scroll the view.
- In conjunction with the changes described under "Messaging" above, all **MessageDropped()** functions have been dropped from the API. **FilterMessageDropped()** is still called, but will be dropped in the future. Switch to BMessageFilter objects.
- The Interface Kit has supporting API (**set_menu_info()**) for the new preference application for menus. Because one of the preferences is the font that's used to display menu items and another is the font size, menu bars can no longer be set to a fixed sized. Therefore, the **B_MENU_BAR_HEIGHT** constant is no more. The BMenuBar class can automatically adjust the size of the menu bar so that it fits the size of the items it displays, given the user's preferences. Applications will have to dynamically adjust the size of other views so that they fit the BMenuBar.
- BMenu's **Track()** function and BPopupMenu's **Go()** have new arguments to support the click-to-open preference and keyboard navigation.
- There's a new preference application and supporting API for the rate and initial delay of repeating keys (see the global functions **set_key_repeat_rate()** and **set_key_repeat_delay()**).
- 1,152-pixel × 900-pixel screen resolutions are now supported. For example, **B_8_BIT_1152x900** is now a screen space that **get_screen_info()** might report.
- The BWindow and BView classes have new versions of their **Convert...()** functions. The new functions are passed BPoint and BRect objects by value and return objects with the converted coordinates. The old versions, which still are in the API, are passed pointers to BPoint and BRect objects, which they convert in place.
- BView's **DrawBitmap()** function has always worked synchronously; it doesn't return until the bitmap image is drawn. There's now a new function, **DrawBitmapAsync()**, that doesn't wait for the Application Server to finish drawing the bitmap; it returns immediately.
- The BControl class has new **KeyDown()** and **MakeFocus()** functions for keyboard navigation. BListView has a new **MakeFocus()** function for the same reason. BListView, and BMenuField, and all BControl classes define navigable views. The BTextControl participates in the new keyboard navigation system, and so has
- BMenuItems that bring up submenus can now also post a message when the submenu comes on-screen.

- The BTextControl object has undergone a number of modifications to make it compatible with the new keyboard navigation system. It has lost its private navigation system and its **GoToNext()** and **GoToPrevious()** functions.
- BTextControl also now lets you set a “modification message” that’s sent when the user first enters or edits text in the field. See the **SetModificationMessage()** function in the class description.
- Some classes have added new functions to set and return parameters previously set only in the constructor. For example, BMenuBar has a **Border()** function, BMenuItem has **SetShortcut()**, BScrollView has **SetBordered()** and **Bordered()**, and BPictureButton has a set of function to return the object’s pictures.
- The BView resizing flag **B_FOLLOW_ALL** has been renamed **B_FOLLOW_ALL_SIDES** to make it clear that only the sides of the parent view are followed. The old constant is still in the header file, but will be dropped for the next release.
- BTextViews respond to **B_SIMPLE_DATA** messages with “text” or “char” entries. See the “Message Protocols” appendix to The Be Book.

The Media Kit and the Midi Kit

The Media and Midi Kits didn’t change much in DR8; look for changes in DR9, when the BStream and BStreamController API will be overhauled and (finally) presented in a usable form. However, both Kits (and the servers and drivers that support them) have had some internal tinkering to enhance stability. In particular, the Audio Server is much less fragile in DR8. Also, the Media Kit’s BAudioSubscriber class can now set the gain of the Audio Server’s loopback control point (the **B_LOOPBACK** device).

One other change that deserves mention is a change in attitude (as opposed to API or implementation): The clique mechanism in the Media Kit’s BSubscriber class should be, in essence, ignored. Always use the **B_SHARED_SUBSCRIBER_ID** constant as the clique value when you subscribe to a media stream. The clique was intended to let a subscriber control access to the stream; it’s not a bad idea, it’s just in the wrong place. Look for the clique mechanism (or similar machinery) to resurface in (possibly) the BStream class in DR9.

The Kernel Kit

In addition to the API changes, listed below, you should be aware that the **libpos.so** library has been subsumed by **libbe.so**, as mentioned above. (This isn’t a Kernel Kit-specific edict, but, in practice, it probably has more of an effect on Kernel Kit-only code, since a BApplication object is not required for all applications and the Kit doesn’t otherwise mandate a BApplication object.)

- The semaphore functions **acquire_sem_count()** and **acquire_sem_timeout()** were replaced by the more general **acquire_sem_etc()**. Similarly, **release_sem_count()** was replaced by **release_sem_etc()**.
- The “etc” business seemed like such a good idea, that it was extended to ports as well: Look for new **read_port_etc()**, **write_port_etc()**, and **port_buffer_size_etc()** functions. The new functions let you set an optional timeout value for operations that could otherwise block forever. No existing functions were harmed in the addition of these new functions.
- To support the new “etc” functions (each of which takes a final bit-field flag argument), a set of four flag values have been added to **<kernel/OS.h>**—the flags used to belong to the Device Kit. Of the four flags, two (**B_CAN_INTERRUPT** and **B_CHECK_PERMISSION**) are intended for device driver writers and can be ignored by all others. The two cogent flags (**B_TIMEOUT** and **B_DO_NOT_RESCHEDULE**) are used by the Kernel Kit’s “etc” functions, although they don’t both apply equally to all of these functions: The timeout flag is used by the blocking functions (**acquire_sem_etc()** and the port functions), while the don’t-reschedule flag is used by **release_sem_etc()**.
- The size and count fields in the **area_info** structure (**ram_size**, **copy_count**, **in_count**, **out_count**) are now **unsigned longs** (they used to be signed). Concomittantly, the *size* arguments to the **create_area()** and **resize_area()** functions are also **unsigned longs**.
- A new single-field structure, **machine_id**, has been added to the Kit. It encodes the local machine’s ID number as two **longs**. Also, new constants have been created that represent the various PPC chips that a BeBox either runs on now, or may run on in the future. Look for the **B_CPU_PPC...** constants in **<kernel/OS.h>**.
- The **system_info** structure has been expanded to include a **machine_id** structure field called **id**, a CPU constant field (**cpu_type**), and a CPU version field (**cpu_revision**).

The Device Kit

The Device Kit now has a BJoystick class corresponding to the joystick ports and three classes—BA2D (analog to digital), BD2A (digital to analog), and BDigitalPort—corresponding to the GeekPort™. *The Be Book* has full documentation. In addition, these changes have been made to the BSerialPort class:

- A new **WaitForInput()** function will block until data arrives to read (or until a timeout expires).
- The **ClearInput()** and **ClearOutput()** functions are now implemented.

Only a couple of changes have been made to the API for kernel-loadable drivers:

- The kernel exports a new **ram_address()** function to drivers. It still exports a set of semaphore functions, but those functions have been made more generally available and are now documented as part of the Kernel Kit.
- There's a new **uninit_driver()** entry point for kernel-loadable drivers. If the driver implements this function, the kernel calls it just before unloading the driver.

The API for graphics card drivers has undergone more extensive changes:

- There are a number of new defined operations to support the Game Kit. Four of the new *op* codes for graphics card drivers correspond to functions defined in the BWindowScreen class of the Game Kit, and another set of four enables the cloning of the graphics card driver so that a BWindowScreen object can have its own copy. See "Developing a Driver for a Graphics Card" in "The Device Kit" chapter of *The Be Book* for details.
- Some arguments in the **graphics_card_info** structure has been moved around. They now match the order in the new **frame_buffer_info** structure, which supports the Game Kit.
- A new flag **B_FRAME_BUFFER_CONTROL** lets a driver report whether it supports custom configurations of the frame buffer.
- There's a new hook function (at index 11) for inverting a rectangle

The Network Kit

Due to popular demand, we broke down and actually documented the Network Kit. This is, in itself, a notable change. But, in addition and at no extra cost, you'll find a whole new topic in the Kit: Mail. The **<net/E-mail.h>** header file contains mail daemon functions, and a mail message class called BMailMessage.

In the socket world, the Be implementations of the **socket.h** and the **netdb.h** functions continue on the road to BSD completeness.

The Support Kit

Get rid of your **B_DECLARE_CLASS_INFO()** and **B_DEFINE_CLASS_INFO()** macro calls. The old class information system has been replaced by an actual real-time info mechanism. Note that the function-like macros (**is_kind_of()**, **is_instance_of()**, **class_name()**, and **cast_as()**) are still valid.

